AD-A100 149 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 9/2
MECHANISM SUFFICIENCY VALIDATION BY ASSIGNMENT.(U)
MAY 81 L J SHIRLEY, R R SCHELL
UNCLASSIFIED NPS52-81-004 NL

1 OF 1
AD A
100149

END
DATE
FILMED
7-81
DTIC

14

NPS52-81-004

NAVAL POSTGRADUATE SCHOOL

Monterey, California

16 MECHANISM SUFFICIENCY VALIDATION BY ASSIGNMENT.

10 Lawrence J. Shirley ~~and~~ Roger R. Schell

11 May 1981 12

Approved for public release; distribution unlimited

Prepared for:

Chief of Naval Research
Arlington, Virginia 22217

1 May 1981

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral J. J. Ekelund
Superintendent

Jack R. Borsting
Provost

The work reported herein was supported in part by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research.

Reproduction of all or part of this report is authorized.

This report was prepared by:

Lawrence J. Shirley
LAWRENCE J. SHIRLEY, LT, USCG

Roger R. Schell
Roger R. Schell, COL, USAF
Associate Professor of
Computer Science

Reviewed by:

G. H. BRADLEY, Chairman
Department of Computer
Science

Released by:

William M. Tolles
WILLIAM M. TOLLES
Dean of Research

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS52-81-004	2. GOVT ACCESSION NO. AD-A100 149	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MECHANISM SUFFICIENCY VALIDATION BY ASSIGNMENT		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Lawrence J. Shirley and Roger R. Schell		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS N0002381R015374
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Supply Systems Command Washington D.C. 20376		12. REPORT DATE May 1981 13. NUMBER OF PAGES 32
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Chief of Naval Research, Arlington, VA 22217		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Presented at the IEEE Symposium on Security and Privacy, April 27-29, 1981

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Security	Protection Domains
Protection Mechanisms	Relational Model
Security Kernel	
Operating Systems	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper introduces a mathematical framework for evaluating the relationship between policies and mechanisms. An evaluation approach called the assignment technique is defined. This technique consists of establishing an assignment between the security classes of information established by policy constraints, and the protection domains, established by the properties of the mechanism. The assignment technique provides a theoretical foundation for assessing the sufficiency of an access control mechanism with respect to

a well formed protection policy. Although this paper presents preliminary results of research, the proposed framework suggests a promising new approach for evaluating the protection mechanisms of existing and proposed systems.

MECHANISM SUFFICIENCY VALIDATION BY ASSIGNMENT

by

Lawrence J. Shirley, Lt. USCG and Roger R. Schell, Col. USAF

Department of Computer Science
Naval Postgraduate School
Monterey, California

ABSTRACT

This paper introduces a mathematical framework for evaluating the relationship between policies and mechanisms. An evaluation approach called the assignment technique is defined. This technique consists of establishing an assignment between the security classes of information established by policy constraints, and the protection domains, established by the properties of the mechanism. The assignment technique provides a theoretical foundation for assessing the sufficiency of an access control mechanism with respect to a well formed protection policy. Although this paper presents preliminary results of research, the proposed framework suggests a promising new approach for evaluating the protection mechanisms of existing and proposed systems.

INTRODUCTION

The suitability of a protection mechanism for any given security policy is not always apparent. This paper presents a theoretical foundation for assessing the sufficiency of an access control mechanism as a means of enforcing a non-discretionary security policy. A technique, termed assignment, establishes a relationship between the information sensitivities of the system entities (partioned according to policy constraints), and dominance domains (inherently established by a protection mechanism). The assignment technique provides a method for mechanism validation, since the results of the assignment can be evaluated to establish whether or not the constraints of the policy are met.

The assignment technique was developed as a means of identifying the limitations of well-formed access control mechanisms. The initial investigation examined the feasibility of using the Multics ring mechanism [13] as a means of enforcing a hierarchical compromise policy. Our basic National Security policy [5] is a well known example. It was established by assignment (as is shown in this paper) that the Multics ring mechanism, of itself, cannot provide this security. On the other hand, it is shown that the Multics ring mechanism does enforce an important form of program integrity policy. This program integrity mechanism can be used to delimit a most privileged set of programs known as the security kernel [11]. The security kernel in turn provides a mechanism sufficient to enforce other security, integrity or access control policies. Thus, with the security kernel technology, the ring mechanism is sufficient for enforcing computer security. By using assignment, we have gained a much better understanding of the capabilities and limitations of a ring protection mechanism, and have introduced a tool for the assessment of other protection mechanisms.

THE PRINCIPLES OF ASSIGNMENT

In order to clearly present the assignment technique we begin with a discussion of the principles of access control. This is necessary because much of the information published in this area appears to be imprecise or even contradictory in nature. Some of the terminology used in this paper may also appear to contradict other authors. These differences and distinctions are intentional and will be discussed in greater detail in an anticipated thesis [14] by Lt. Shirley. This paper merely addresses the basic framework which we choose for our discussion.

Lattice Security Policies

A security policy is based upon external laws, rules, regulations and other mandates that establish what access to information is to be permitted. We choose as our universe of discourse the lattice security policies as identified by Walters [15] and later also described by Denning [5]. These universally bounded lattice structures consist of finite, partially ordered sets of access classes, each having a least upper and greatest lower bound. This class of policies encompasses many (if not all) practical policies. Such policies are of primary interest to National Defense because all non-discretionary security policies can be represented as a lattice policy. To be effective, such policies must clearly establish an access class for all system entities, i.e., subjects (the active entities) and objects (the passive entities that may be referenced by a subject). Furthermore, the policy must identify all permissible access relations between the subjects and objects of various equivalence classes. If a policy were not able to meet these two requirements, the enforcement of the policy could not be evaluated.

Note that we distinguish between processes and subjects in this paper. This is necessary because of the ambiguity that might result without the distinct notion of a subject as a process-domain pair [9, 12], particularly when we present a formalized definition of a domain.

Access Relations

Any specific policy will distinguish one or more distinct access relations between subjects and objects. These are typically mirrored in the "access mode" of the corresponding protection mechanism.

Two generic access modes are sufficient for a general discussion of the principles and policies discussed in this paper. These are [7] "observe" (the ability to observe information) and "modify" (the ability to modify information). Other primitive access modes are generally just a finer granularity of observation and modification priviledges.

The enforcement of a policy is fundamentally limited by the system's granularity of access. Policies that prescribe distinctions not recognized by the access control mechanisms must be enforced in an overly restrictive manner or ignored. For example, a policy addressing a concatenation access relation cannot be precisely enforced on a system that does not recognize some form of append access mode.

The granularity of access control within a system is dependent upon the ability to distinguish attributes of subjects and objects and upon the variety of access modes available. The primitive access modes are associated with the design of the system, including the protection mechanisms, and designate the associated rights obtained by an access request.

An access relation is a tuple (subject, access mode, object). This tuple signifies that a relation between the subject and object exist such that the subject is permitted to access the object with all the privileges associated with the access mode. The problem of information security may generally be expressed as the problem of permitting the existence of only those access relations that in no way violate any of the applicable systems policies.

Basic National Security Policy Example

The basic National Security policy is a simple lattice policy. The policy defines entities as members of one of four hierarchical access classes (UNCLASSIFIED, CONFIDENTIAL, SECRET, TOP SECRET). The greatest lower bound is UNCLASSIFIED and the least upper bound is TOP SECRET. Figure 1(A) represents this lattice structure.

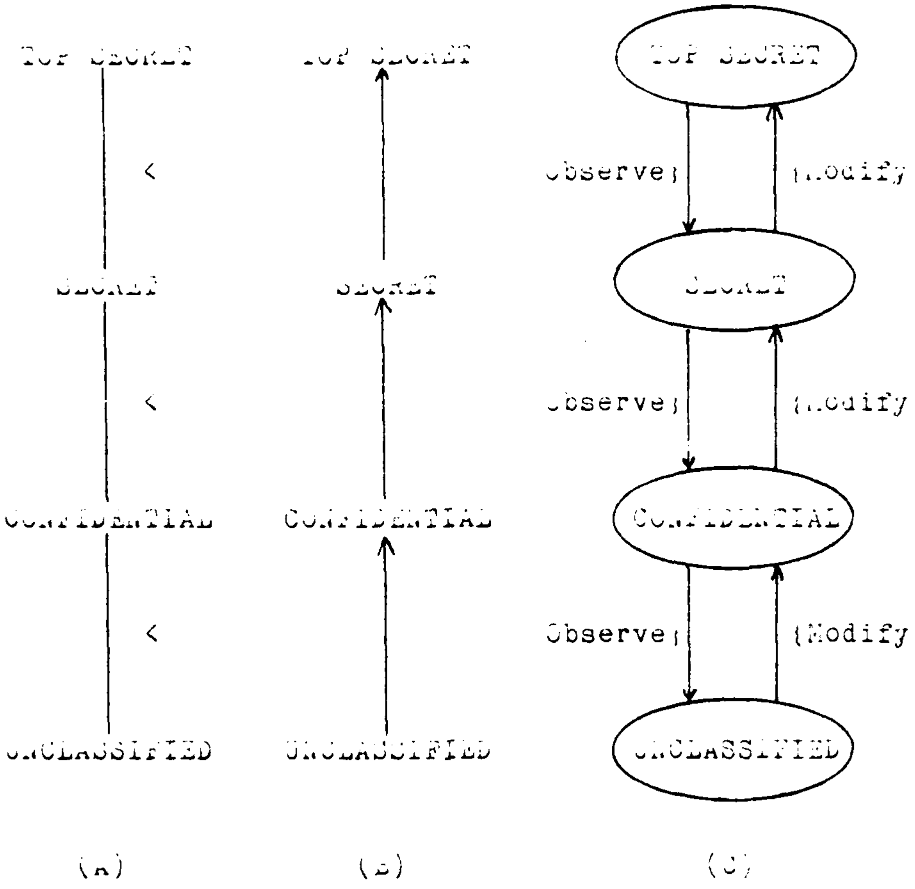


Figure 1

Figure 1(B) shows the information flow characteristics of this lattice policy [5]. This information transfer path [15] can be analyzed with respect to permissible access relations.

Based on this analysis of the permissible access relations between (subjects and objects with) the various access classes, we derive an alternative illustration form that is convenient for our analysis. Figure 1(C) illustrates the basic National Security policy using this form. Note that a node represents an equivalence class of entities all of which have the same access class. A directed arc represents the permissible access relations from a subject of the source equivalence class to objects of the destination equivalence class. Transitivity of access relations is not shown but is assumed.

Recall that a system is "secure" if there are no access relations that violate any applicable policy. The Simple Security Condition [1] states that if observe access is permitted, then the access class of the subject is greater than or equal to the access class of the object. The "Confinement Property" -- historically known by the less descriptive name of * - Property [1] -- states that if modify access is permitted, then the access class of the subject is less than or equal to the access class of the object. We can see that Figure 1(b) is derived directly from these two properties.

Access Domains

So far, we have concentrated on the properties of policies. We now examine the properties of the protection mechanisms used to enforce security policies. The principle notion we use is that of an access domain.

An access domain Δ, is a tuple, (a_1, a_2, ..., a_i, ..., a_n), where n is the number of primitive access modes in the system, and a_i is the set of all objects, { o_1, o_2, ..., o_j, ..., o_m } which a process executing in domain Δ may access by access mode "i". We speak of an (access mode)-domain as the set of objects which a process executing in that domain has the right to access according to that particular access mode.

Consider the following two domains:

Δ_1 : (observe(O): {A,B,C}, modify(M):{A,D,E})

Δ_2 : (O: {A,C,D,E}, M:{ ϕ })

The observe-domain of Δ_1 (denoted as $O\Delta_1$) is objects A, B, and C. The modify-domain $M\Delta_2$ is empty.

A set of dominance domains are implicitly established by the system's protection mechanisms. The dominance domains are not associated with any particularization of processes and objects, but rather dominate all the domains that may occur in the system.

Dominance domains may be uniquely labeled for convenience. In the Multics system, for example, the dominance domains established by the ring mechanism were known as rings and were labeled by ring numbers. Schrceder's protection mechanism also uses numbers as labels for dominance domains [12].

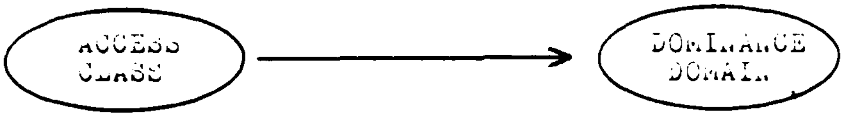
We say that Δ_1 dominates ($\propto$) Δ_2 iff for each a_i, $a_i\Delta_2 \subseteq a_i\Delta_1$. The systems protection mechanism then, establishes a set of dominance domains which we can use for validation of protection mechanisms. Because these domains dominate all other domains that may occur in the system, if we can show that our policy holds for these domains, we have shown that it holds for the system.

In this paper, we choose to consider only protection mechanisms which establish a universally bounded lattice of dominance domains. Such mechanisms represent an interesting subset of protection mechanisms and provide simplicity in this discussion.

The Assignment Technique

Assignment is the establishment of a relationship between two entities such that the first entity is "assigned to" the second entity. Mathematically, the term assignment is not significant. One could easily have said that entity 1 is related to entity 2. Intuitively, however, assignment is associated with the connotation "to fix authoritatively" which precisely signifies our notion of this process.

Assignment may be denoted by a graph from the first entity to the second as follows:



"is assigned to"

Assignment does not alter either entity. Rather, a relationship between the entities is established which can be expressed in the form of a tuple as follows:

ACCESS CLASS	DOMINANCE DOMAIN

"is assigned to"

Regardless of the means of representation, assignment is merely the act of associating an entity or set of entities with some other entity or set of entities.

The essence of the assignment technique is relatively simple. First of all, consider the nature of a lattice security policy. Such a policy partitions the objects of a system into a lattice of equivalence classes. Each equivalence class can be thought of as an entity subject to assignment.

Then consider a mechanism, which establishes a lattice of dominance domains. Each of these domains can also be thought of as an entity subject to assignment.

Since an assignment can be established between any two entities, we can make an assignment between the equivalence classes established by a lattice security policy and the dominance domains that are established by some protection mechanism. We then validate that (for this assignment) the mechanism is sufficient to support that policy. This determination is made by examining the set of access relations that the mechanism permits, and testing for possible violations of the policy.

We are now ready to illustrate how we may use this assignment technique to evaluate protection mechanisms used in the design of secure computer systems.

APPLICATIONS OF ASSIGNMENT

The usefulness of the assignment technique appears to be rather far reaching in scope. Research currently underway is investigating a number of possibilities. This paper addresses only a few of the possible applications. The authors wholeheartedly invite the reader to suggest areas of further research. Additionally, comments, opinions, and research findings related to the assignment technique are solicited.

Multics Ring Mechanism Assignments

The question of the sufficiency of the Multics Ring Mechanism for enforcement of the basic National Security policy was the initial problem that prompted the current research effort and led to the formulation of the assignment technique. It is appropriate then, that this paper present this analysis as an introductory application of simple assignment.

Compromise Policy. As stated previously in this paper, the basic National Security policy is a simple lattice security policy. Figure 1(C) illustrates this policy.

The dominance domains of the Multics Ring mechanism are most frequently shown as concentric rings numbered in increasing integer order from the innermost ring or the kernel. The kernel is generally assigned ring number 0. For simplicity, we only show a system with rings 0 thru 3 in this analysis. Other ring numbers will produce similar results.

We begin by assigning the least upper bound of our lattice to ring 0. The assignment produced is illustrated in Figure 2.

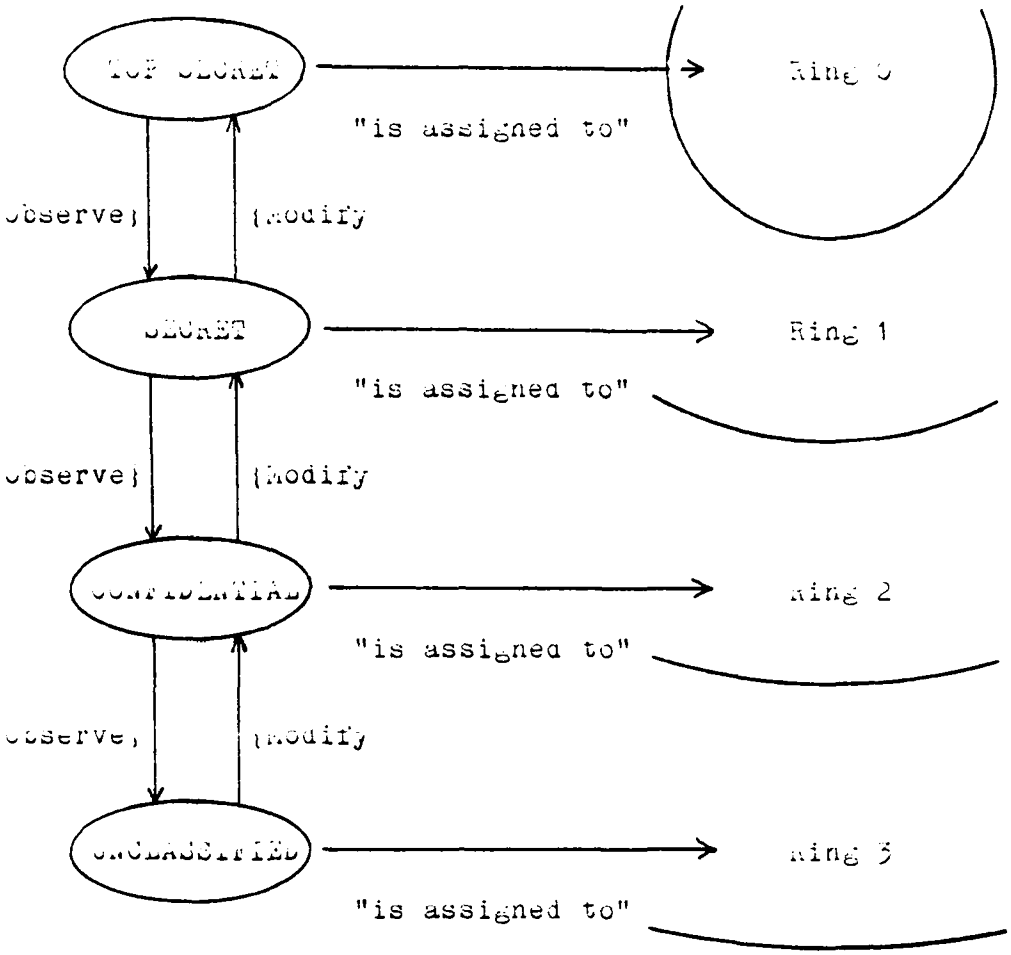


Figure 2

Now we must examine the access relations permitted by the mechanism and test for possible violations of the policy. In order to do so, we must examine the nature of the Multics Ring Mechanism more closely.

The Multics Ring Mechanism determines the authorized access of a process by means of the current ring number (r). Thus a process which is executing in ring number 1 would need to be cleared for at least SECRET information according to our assignment scheme.

The Multics Ring Mechanism discriminates among objects by means of a ring bracket. The ring bracket is a 3 - tuple (R1, R2, R3) where R1, R2 and R3 are ring numbers and $R1 \leq R2 \leq R3$. Access to objects is restricted such that the current ring of execution must be less than or equal to R2 to observe information and less than or equal to R1 to modify information. Figure 3 shows characteristics of the ring brackets both in terms of the access modes used in this paper and the access modes used in Multics.

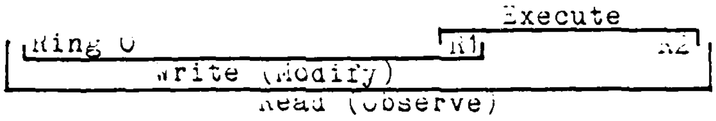


Figure 3

Consider then an object that is classified as SECRET. Such an object must be assigned a ring bracket such that it may be observed by processes in ring 0 and ring 1 only. R2 must therefore be 1. A problem now becomes apparent. No matter what value we choose for R1, we are faced with a contradiction. If R1 is 0 or 1 then TOP SECRET processes may modify SECRET files violating the Confinement Property. If R1 is greater than 1, the restrictions of the ring mechanism would be violated (viz., $R1 > R2$). Therefore, we can conclude that this assignment is not acceptable.

Consider now the only other potential assignment scheme where the greatest lower bound of our lattice is assigned to ring 0. The assignment produced is shown in Figure 4.

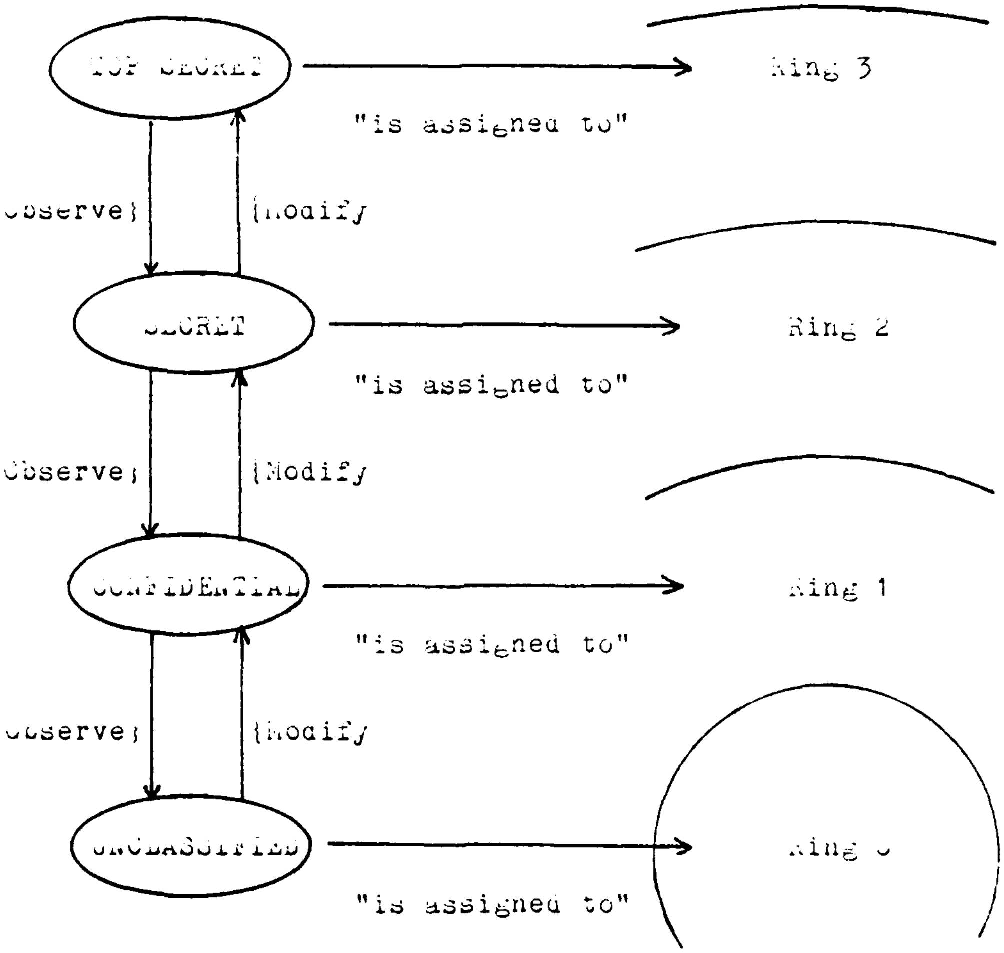


Figure 4

We now attempt to assign ring brackets to an object classified SECRET. A problem occurs immediately. We want processes executing in ring 2 to be able to observe our SECRET objects, but then a process in ring 0, that is UNCLASSIFIED, will also be able to observe our object. The Simple Security Condition cannot be enforced with this assignment so the assignment scheme is not feasible.

Since neither of these assignments are acceptable, and shifting the ring assignments numerically would yield similar results, we can see that no assignment will be acceptable. Therefore, the Multics Ring Mechanism is not sufficient to enforce the basic National Security policy for compromise.

The basic National Integrity policy [2] is the dual of the basic National Security policy. Whereas the security policy is concerned with the unauthorized observation of information or compromise, the integrity policy is concerned with the unauthorized modification of information or subversion. The assignment technique shows us that the Multics Ring Mechanism is not sufficient to enforce this dual policy either.

The Multics Ring Mechanism is not sufficient to enforce the basic National Security policy nor the basic National Integrity policy. However, a Multics Security Kernel has been designed [15] that is sufficient to support both of these policies. This may seem to be a contradiction, but it is not. The confusion is dissipated when one asks the question, "What form of policy does the Multics Ring Mechanism support?"

Program Integrity Policy. The notion of a program integrity policy stems from the desire to prohibit modification of executable programs by less trustworthy subjects. In the general sense, we wish to ensure that our more sensitive programs are "tamperproof." Unlike a strict integrity policy, however, program integrity is not concerned with the issue of general observation of information. Rather, program integrity deals only with execution and modification. In this case, we refine the access mode "observe" to that of "read/execute" access mode, taken in the sense of the general vernacular.

A program integrity policy must consider two issues. First, each entity within the system must have a program integrity access class, designated PI, assigned to it. Second, the ordering of program integrity access classes must be fixed according to the constraints of the policy maker. Once these issues are resolved, we may guarantee that no direct threat is possible by enforcement of the following condition:

> *Simple Program Integrity Condition* : If a subject has "modify" access to an object, then the program integrity of the subject is greater than or equal to the program integrity of the object.

Because program integrity policies are concerned with the execution issue, indirect modification of information is not strictly prohibited. This provides a certain degree of flexibility but also produces a certain amount of risk [6]. Confinement of execution helps to reduce the risk of such an indirect threat. The indirect threat occurs when a subject executes a program that has been modified by another less trustworthy subject. We can further see the usefulness of confinement in a program integrity policy by noting that this property supports the use of library function. In a manner directly analogous to that for the

National Integrity policy [2], we define the confinement property for program integrity as follows :

> Program Integrity Confinement Property : If a subject has execute access to an object then the program integrity of the object is greater than or equal to the program integrity of the subject.

The characteristics of an example program integrity policy in terms of access modes is shown in Figure 5. Such a policy is inherently a lattice policy.

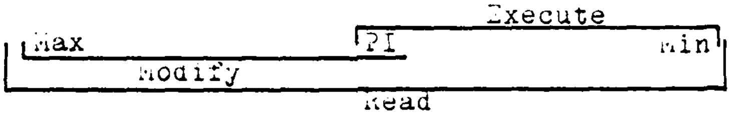


Figure 5

Consider now a specific program integrity policy. According to this policy, entities are partitioned into one of four access classes designated as User, Supervisor, Utility or Kernel. The sensitivity of these access classes is specified as : Kernel > Supervisor > Utility > User. We then consider an assignment to a Multics ring structure as shown in Figure 6.

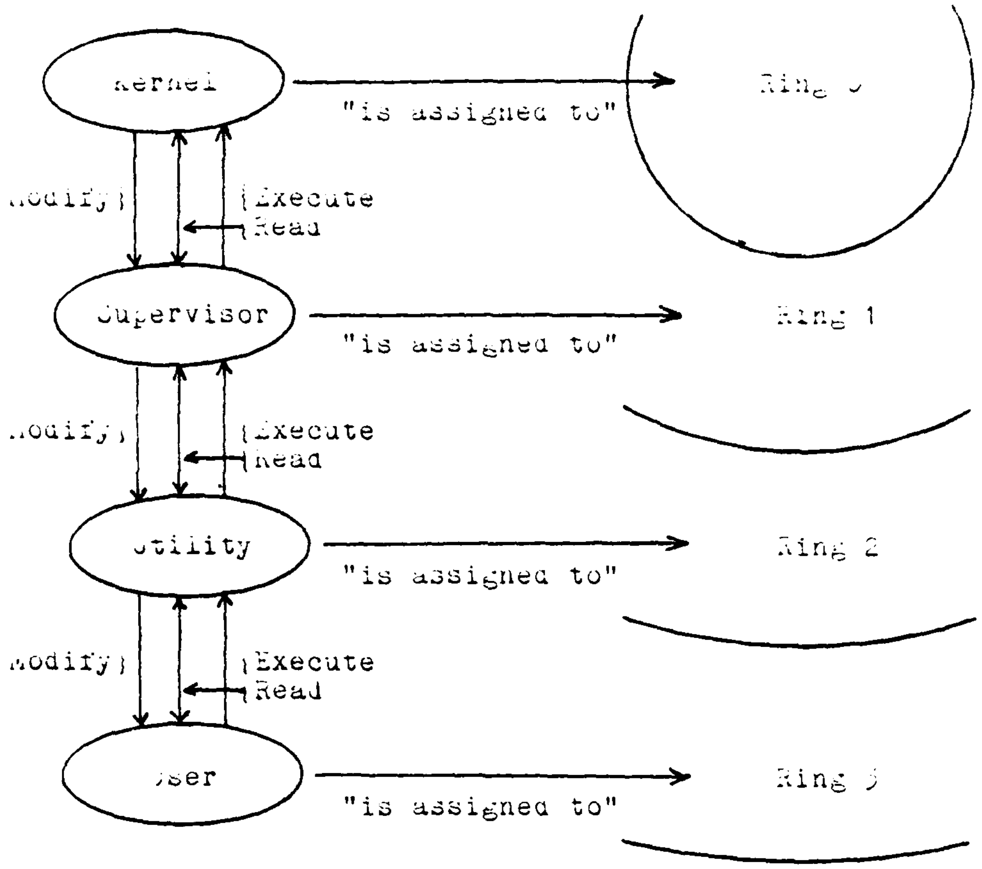


Figure 6

Recalling the characteristics of ring brackets shown in Figure 5, we designate "Max" as Ring 0, the program integrity access class "P1" as R1 and "Min" as R2. We note that for *this* policy any choice for R2 greater than or equal to R1 will do. A more sophisticated policy requiring the notion of a "gate" is beyond the scope of this paper.

We now examine the access relations permitted by the mechanism and test for possible violations of the policy. From this examination, we can see that the read-domains, the modify-domains and the execute-domains for each ring (Ring 0 thru Ring 3) permit all valid access relations to occur, and prohibit the occurrence of any invalid access relation with respect to this policy. So for this assignment, no violations are possible. Therefore, we have shown that the Multics Ring mechanism is sufficient to support this Program Integrity policy.

This issue of what form of protection the Multics Ring mechanism provides, appears to be precisely the issue that Wulf, Jones and the other designers of the "HYDRA" system were attempting to understand [16]. They introduce their discussion by first saying :

> "Protection is, in our view, a mechanism." [16]

Their discussion then proceeds to make the following general statement relative to the Multics rings:

> "Our rejection of hierarchical system structures and especially ones which employ a single hierarchical relation for all aspects of system interaction, is also, in part, a consequence of the distinction between protection and security. A failure to distinguish these issues coupled with a strict hierarchical structure leads inevitably to a succession of increasingly privileged system components, and ultimately to a "most privileged" one, which gain their privilege exclusively by virtue of their position in the hierarchy. Such structures are inherently wrong ..." [16]

Had the assignment technique been available to the authors of the above statement, they would have been afforded a means of expressing their views more precisely than the ambiguous phrase "inherently wrong". The assignment technique provides a precise means for clearly formulating such an observation and evaluating its validity. As shown, and in agreement with Wulf's statement, the Multics Ring Mechanism is "inherently wrong" with respect to compromise policies. On the other hand, the Multics Ring Mechanism is just "right" as a means of enforcing a program integrity policy or assisting in the enforcement of the system's non-hierarchical security policies (viz., via Security Kernels).

Other Ring Mechanisms

The Multics Ring Mechanism is by no means the only form of Ring Mechanism. By altering the requirements of the Ring brackets and the need for a Gate Keeper, one can contemplate adapting the ring mechanisms to meet other simple hierarchical policies.

Consider using the assignment shown in Figure 2, but altering the means of discrimination among objects such that the Ring bracket is a singleton (R1). Following the rules shown in Figure 7, we can adapt this ring mechanism to enforce the basic National Security policy.

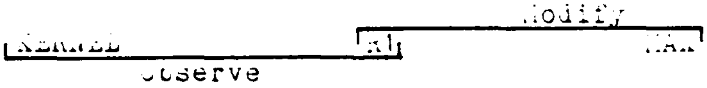


Figure 7

Similarly, Figure 8 shows the rules necessary for the same assignment as shown in Figure 2 to adapt this ring mechanism to meet the basic National Integrity policy. Examining Figure 7 and Figure 8, the dual nature of these two policies is apparent.

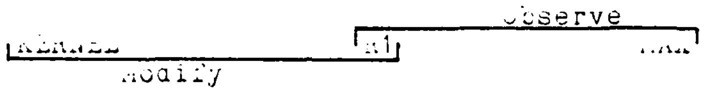


Figure 8

To be sure, these brief suggestions do not completely characterize a practical protection mechanism. However, it appears that ring mechanisms are adaptable for the enforcement of various simple hierarchical policies.

Capability Mechanisms

Considerable effort is currently underway to provide Provably Secure Operating Systems based upon the capability mechanism [8,10]. It is important to examine what form of protection capabilities actually provide.

Capability mechanisms primarily establish two dominance domains which are enforced by the system hardware. One domain consists of capabilities, and the other is objects that are not capabilities such as segments and directories. A process takes no note of these dominance domains, however, because all processes have access to capabilities as well as other types of objects. So with respect to a process, the capability mechanism provides no inherent partitioning of the system entities at all. In fact, in trying to determine the structure of dominance domains for non-capability objects, we encounter a veritable "spaghetti bowl" of domains, devoid of any inherent, unifying structure. Thus a capability mechanism is of itself not sufficient for the enforcement of any non-discretionary policy.

This is not to say that a capability mechanism is not useful. For example, the mechanism can protect a security kernel in much the same way as rings protect the kernel in the Multics design.

CONCLUSIONS

Assignment has been shown to be a useful technique in evaluating the sufficiency of a mechanism to enforce a security policy. This technique is based upon a formalized notion of domains and the lattice nature of security policies.

This method provides considerable insight into the nature of access control. Characterizing a subject as a process-domain pair, we observe that non-discretionary protection is dependent only upon the dominance domains established by the systems mechanisms and the access relations between these domains. The nature of the computation is irrelevant. Furthermore, one can observe that any protection policy can only be implemented on a computer system which has some form of system isolation prohibiting the users from altering the system's isolation method.

This paper presents an introduction to assignment, and several simple examples have been investigated. Considerable research effort is still necessary. Of particular interest is the use of the assignment technique as a guide in the construction of new mechanisms to meet classes of policies of broad interest. Assignment research has already provided considerable insight to the nature of security enforcement, providing a means of formally presenting the characteristics of mechanisms and policies. Mechanisms can be categorized by the type of enforcement that they provide thus giving the system's designer a tool for selection of mechanisms to meet the security objectives of the system in question.

REFERENCES

[1] Bell, D. E. and LaPadula, L. J., Secure Computer System: Unified Exposition and Multics Implementation, MITRE Corporation Report ESD-TR-75-306, (AD-A023 588), March 1976.

[2] Biba, K. J., Integrity Considerations for Secure Computer Systems, MITRE Corporation Report, ESD-TR-76-372, (AD-A039 324), April 1977.

[3] Denning, D. E., Secure Information Flow in Computer Systems, Ph. D. Thesis, Purdue University, May 1975.

[4] Denning, D. E., "A Lattice Model of Secure Information Flow", Communications of the ACM, Vol. 19, p. 236 - 243, May 1976.

[5] Department of Defense, DoD 5200.1R, DoD Information Security Program Requirements.

[6] Feirtag, R. J. and Neuman, P. G., "The Foundations of a Provably Secure Operating System (PSOS)", AFIPS National Computer Conference Conference, 1979, p. 329 - 334.

[7] Grohn, M. J., A Model of a Protected Data Management System, Canadian Commercial Corporation Report, ESD-TR-76-289, (AD-A035 256), June 1976, p. 45.

[8] Jones, A. K., "Protection Mechanism Models: Their Usefulness", in Foundations of Secure Computations, edited by R. A. DeMillo and others, p. 237 - 254, New York: Academic Press, 1978.

[9] Lampson, B. W., "Protection", Proceedings Fifth Annual Conference on Information Sciences and Systems, Princeton University, p. 437 - 443, March 1971.

[10] Neumann, P. G., Robinson, L., Levitt, K. N., A Provably Secure Operating System, Stanford Research Institute Report, (AD-A080 001), June 1975.

[11] Schell, R. R., "Security Kernels: A Methodical Design of System Security," USE Technical papers (Spring Conference, 1979), March 1979, p. 245-250.

[12] Schroeder, M. D., Cooperation of Mutually Suspicious Subsystems in a Computer Utility, AD-750 173, Ph. D. Thesis, Massachusetts Institute of Technology, September 1972.

[13] Schroeder, M. D., Clark, D. D., and Saltzer, J. H., "The Multics Kernel Design Project", Proceedings of Sixth ACM Symposium on Operating Systems Principles, November 1977, p. 43-56.

[14] Shirley, L. J., Non-Discretionary Security Validation by Assignment, Masters Thesis, Naval Postgraduate School, in preparation.

[15] Walter, K. G. and others, Primitive Models for Computer Security, Case Western Reserve University Report, ESD-TR-74-117, (AD-778 467), January 1974.

[16] Wulf W. and others, Hydra: The Kernel of a Multiprocessor Operating System, Carnegie-Mellon University Report, AFOSR-TR-73-1079, (AD-762 514), June 1973, p. 10.

INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center — 2
Cameron Station
Alexandria, Virginia 22314

2. Library, Code 0142 — 2
Naval Post Graduate School
Monterey, California 93940

3. Office of Research Administration — 1
Code 012A
Naval Post Graduate School
Monterey, California 93940

4. Department Chairman, Code 52 — 30
Department of Computer Science
Naval Postgraduate School
Montery, California 93940

5. COL Roger R. Schell, Code 52Cj — 10
Department of Computer Science
Naval Postgraduate School
Montery, California 93940

6. Lyle A. Cox, Jr., Code 52Cl — 4
Department of Computer Science
Naval Postgraduate School
Montery, California 93940

7. Commandant, U. S. Coast Guard (G-FIS) — 2
2100 2nd St. S.W.
Washington, D.C. 20590

8. Commandant, U. S. Coast Guard (G-PTE) — 2
2100 2nd St. S.W.
Washington, D.C. 20590

9. Lawrence J. Shirley — 10
SMC Box 2081
Naval Postgraduate School
Montery, California 93940

10. James P. Anderson — 1
Box 42
Fort Washington, Pa. 19034

11. Terry C. Arnold 1
2555 Camino Del Rio South
Suite 250
P. O. Box 20217
San Diego, Ca. 92120

12. J. W. Freeman 1
Computer Sciences Corp.
803 West Broad Street
Falls Church, Virginia 22046
Attn: Ingrid Hues

13. Bruce Goldstein, Pres. 1
Executect
1100 Gough Street, Suite 8F
San Francisco, Ca. 94109

14. Daniel G. Hoblyer 1
The Aerospace Corporation
P. O. Box 92957
Los Angles, Ca. 90009

15. Marvin Schaefer 1
System Development Corporation
2500 Colorado Avenue
Santa Monica, Ca. 90406

16. John P. Schill 1
Code 8321 Concept Development Branch
C3 I Sys Dept
Naval Ocean Systems Center
San Diego, Ca. 92152

17. Peter S. Tasker 1
The Mitre Corporation
P. O. Box 208 MS/B-352
Bedford, Mass. 01730

18. Rein Turn 1
California State University, Northridge
Department of Computer Science
18111 Nordhoff Street
Northridge, Ca. 91330

19. Kyle L. White 1
P. O. Box 1621
Vandenburg AFB, Ca. 93437

20. John Woodward 1
The Mitre Corporation
P. O. Box 208
Bedford, Mass. 01730

21. Kathryn Heniger, Code 7503 1
Naval Research Lab
Washington, D. C. 20375

22. Joel Trimble, Code 221 1
Office of Naval Research
800 North Quincy
Arlington, Va. 22217

23. Chief of Naval Research 1
Arlington, Va. 22217

24. Carl Landwehr 1
Code 7522
Naval Research Laboratory
Washington, D. C. 20375

25. Steven B. Lipner 1
Digital Equipment Corp.
ML5-2/E41
146 Main Street
Maynard, Mass. 01754

DATE

ILMED

-8